## COP2806C Module 1 Programming Assignment Part 1 - Introduction to Spring

In this assignment we will implement the Hello World program from Ch. 1 of our Spring textbook. Follow the instructions below to create the project and execute the application, then submit your project folder to the GitHub classroom repo;  include a screen snip of the final step showing the Hello World message in your repo submission. There is no Canvas submission for this assignment,

We will be working from the command line; while professional IDEs would make this easier, we need to become familiar with command line operations as we frequently need to use them for testing and automation tasks. You can use an IDE to edit the files if desired, but the files must be located in the correct locations based on the instructions provided.

I would recommend that you use the Horizon system for this assignment as all software is installed and preconfigured. If you use Horizon, download this instruction document on that system to insure you can copy and paste as necessary.

You will need Gradle, Java 17, and an Internet connection for this assignment.

1. Start by creating a project folder named **isf6**. On Horizon you can create this folder in your Downloads folder or the Documents folder.

```
C:\Users\YourUserID\DownLoads>mkdir isf6
```

2. Move to the isf6 folder.

```
C:\Users\YourUserID\DownLoads>cd isf6
```

3. Run **gradle init** to set up the gradle environment in this folder:

```
C:\Users\YourUserID\DownLoads\isf6>gradle init
```

4. Using your favorite text editor, modify the **build.gradle** file in this folder that was created by the previous command as shown in Listing 1-1 in the text. Refer to the text for an explanation of the contents of this file. Note that the textbook states "As this book is being written, Spring 6 has not been released" -- but Spring 6 has now been released (November 2022).

```
apply plugin: 'java'
sourceCompatibility = 17
targetCompatibility = 17
ext {
    springFrameworkVersion = "6.0.0-M4"
    testNgVersion = "7.6.1"
}
allprojects {
    apply plugin: 'java'
    repositories {
        maven {
            url "https://repo.spring.io/milestone"
        }
        maven {
            url "https://repo.spring.io/snapshot"
            mavenContent {
                snapshotsOnly()
            }
        }
        mavenCentral()
    }
    dependencies {
        implementation \
        "ch.qos.logback:logback-classic:1.2.11"
        testImplementation \
        "org.testng:testng:$testNgVersion"
    }
    test {
        useTestNG()
    }
}
```

5. Modify the **settings.gradle** file in this folder as shown in Listing 1-2 in the text:

```
pluginManagement {
    repositories {
        maven {
            url "https://repo.spring.io/milestone"
        }
        maven {
            url "https://repo.spring.io/snapshot"
            mavenContent {
                snapshotsOnly()
            }
        }
        gradlePluginPortal()
    }
}
rootProject.name = 'isf6'
include 'chapter01-hello-world'
```

6. Create a subfolder named **hello-world-chapter-01**.

```
C:\Users\YourUserID\DownLoads\isf6>mkdir hello-world-chapter-01
```

7. Move to **hello-world-chapter-01** subfolder.

```
C:\Users\YourUserID\DownLoads\isf6>cd hello-world-chapter-01
```

8. Create a **build.gradle** file in this folder as shown in Listing 1-3 in the text:

```
plugins {
    id "application"
}
dependencies {
    implementation \
    "org.springframework:spring-core:$springFrameworkVersion"
    implementation \
    "org.springframework:spring-context:$springFrameworkVersion"
    implementation \
    "org.springframework:spring-test:$springFrameworkVersion"
}
application {
    mainClass.set("chapter01.Application")
}
```

7. Create the source code folder structure shown below (the path through the Downloads folder has been omitted and replaced with an ellipsis "..." in the following instructions for brevity). You can create a multi-level folder hierarchy by using a single mkdir command.

```
C:\...\isf6\hello-world-chapter-01>mkdir src\main\java\chapter01
```

8. Move to the **chapter01** folder and create the following Java source code files:

**MessageService.java (Listing 1-4)**

```
package chapter01;

public interface MessageService {
  public String getMessage();
}
```

**HelloService.java (Listing 1-5)**

```
package chapter01;

public class HelloService
  implements MessageService {
  @Override
  public String getMessage() {
    return "Hello, world!";
  }
}
```

**Application.java (Listing 1-6)**

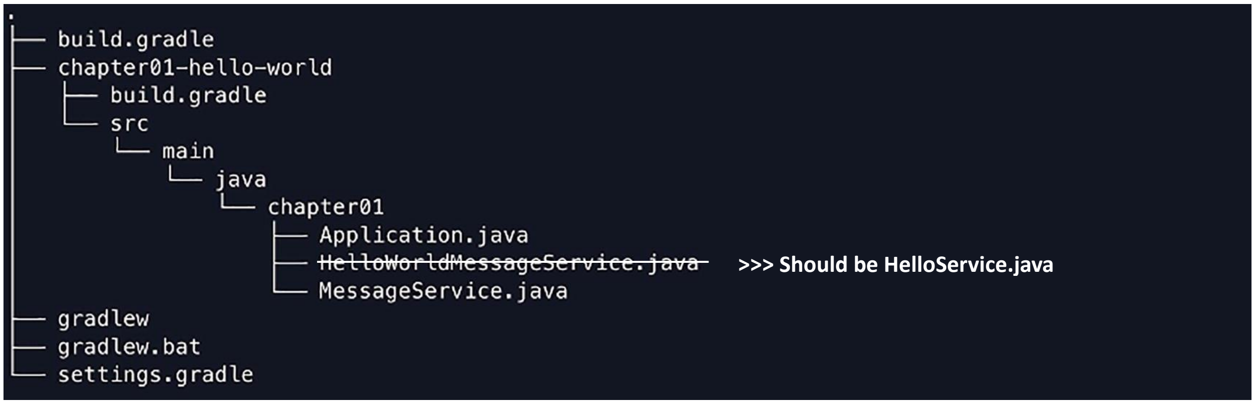
```
package chapter01;
```

```
import org.springframework.context.annotation.*;

@Configuration
public class Application {
  @Bean
  MessageService helloWorldMessageService() {
    return new HelloService();
  }
  public static void main(String[] args) {
    var context =
      new AnnotationConfigApplicationContext(Application.class);
    var service =
      context.getBean(MessageService.class);
    System.out.println(service.getMessage());
  }
}
```

Your project directory tree should now look like that shown in Figure 1-3 from the text (Note that the text images shows the file "HelloWorldMessageService.java" in the source folder, this is not correct; it should be "HelloService.java").

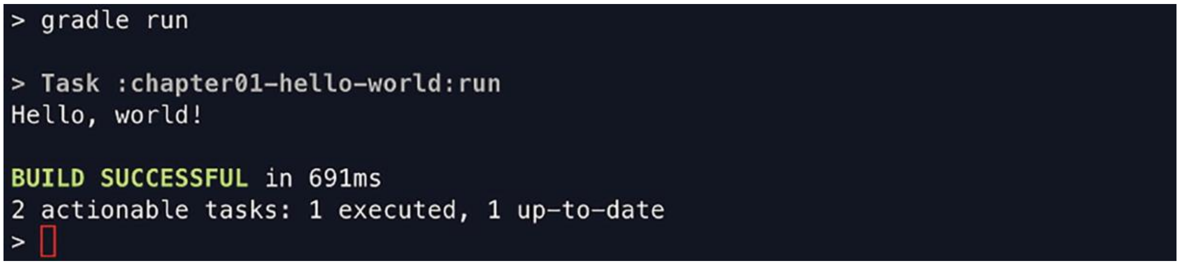
**Figure 1-3 The Directory Tree So Far**



```
.
├── build.gradle
├── chapter01-hello-world
│   ├── build.gradle
│   └── src
│       └── main
│           └── java
│               └── chapter01
│                   ├── Application.java
│                   ├── ~~HelloWorldMessageService.java~~   >>> Should be HelloService.java
│                   └── MessageService.java
├── gradlew
├── gradlew.bat
└── settings.gradle
```

9. Execute the command gradle run in the isf6 folder as shown in Figure 1-4 of the text to build and execute the project.

**Figure 1-4 Executing the chapter01-hello-world project**



```
> gradle run

> Task :chapter01-hello-world:run
Hello, world!

BUILD SUCCESSFUL in 691ms
2 actionable tasks: 1 executed, 1 up-to-date
> 
```